

Table based KNN for combining Text Segmentation with Text Categorization

Taeho Jo
President
Alpha AI Research
Cheongju, South Korea
tjo018@naver.com

Abstract—In this research, we propose the table based KNN version for automating the text segmentation with its combination with the text classification. In the previous work, we proposed the table based KNN version as an approach to the text segmentation but were required to provide the domain as a tag for performing the task. In this research, we adopt the KNN algorithm which receives a table instead of a numerical vector as the approach to the text segmentation and the text classification. In the proposed system, only a text without its domain is given as the input, the domain is automatically assigned to the text through the text classification, and each paragraph pair is classified into boundary or continuance by the classifier which corresponds to the domain. This research is intended to automate completely the text segmentation by introducing the text classification, and to improve the performance of both tasks, using the table based KNN algorithm.

I. INTRODUCTION

The text segmentation is defined as the process of segmenting a text into subtexts based on its contents. Because a boundary is set between paragraphs, the text segmentation is converted into a binary classification of an adjacent paragraph pair into boundary or continuance. The process of text segmentation is to partition a text into paragraphs, generate adjacent paragraph pairs by sliding a two-sized window on them, and classify them into boundary or continuance. A boundary is established between paragraphs in a pair which is classified into boundary, and a text is segmented into subtexts by the established boundaries. The classification task which is mapped from the text segmentation is a domain dependent task where an identical entity is classified differently, so a domain should be presented together with a text for doing the text segmentation.

This research is motivated by the requirement of presenting a domain for doing the text segmentation. It is given as an independent classification tasks as many as domains; a classifier should be allocated to each domain independently of others. We need to label the input text manually with the domain for nominating a classifier which corresponds to a domain. Only a text is presented without its own domain by assigning a domain to a text automatically through the text classification. To solve the problem, the text segmentation which is covered in this research relates to the text classification.

The idea of this research is to connect the text segmentation with the text classification and apply the table based KNN version to both tasks. We define the similarity metric between tables which represent both words and texts and use it for modifying the KNN algorithm into the table-based version. It is applied to the text classification which assigns a domain automatically to an input text. The text segmentation is mapped into a domain dependent binary classification, a suitable classifier is nominated by the domain which is obtained by the text classification, and each of adjacent paragraph pairs is classified by the table based KNN version. A boundary is put between paragraphs in each pair which is classified into boundary, and the text is segmented into subtexts, following the boundaries.

Let us mention some benefits from this research. The problems in representing texts into numerical vectors are solved by representing them into tables. The performance of the text segmentation and the text classification is improved by adopting the table based KNN algorithm. The process of deciding a domain to a text is automated by connecting the text segmentation with the text classification. The chance of modifying other machine learning algorithms is provided by modifying KNN algorithm so.

Let us mention the organization of this paper. In Section II, we explore the previous works which are relevant to this study. In Section III, we describe in detail what is proposed in this research. In Section IV, we mention the significances of this research and the remaining tasks. This paper is composed of the four sections.

II. PREVIOUS WORKS

This section is concerned with previous works which are relevant to this research. It is intended to connect the text segmentation with the text classification for automating the classifier selection. The direction of exploring previous works is to find cases of applying the proposed version of KNN algorithm to the text classification and the text segmentation and proposing that a text is represented into a table for improving the classification performance. The distinguishable point of this research is that the text segmentation is connected to the text classification for implementing the complete text segmentation system. This section is intended

to explore previous works for providing background for this research.

Let us explore the previous works on applying the modified version of KNN algorithm to text categorization. In 2017, it was initially proposed that the table-based version of KNN algorithm should be applied to the text classification by Jo [4]. In 2018, the modified KNN algorithm was validated in the task by Jo [6]. The journal article which describes in detail the modified KNN algorithm and its application to the text classification is finally prepared for its publication by Jo [8]. In the pervious works, better performances were observed in applying the proposed version of the KNN algorithm to the text classification.

Let us explore the previous works on applying the modified version of KNN algorithm to text segmentation. In 2017, it was initially proposed that the table-based version of KNN algorithm should be applied to the text segmentation by Jo [5]. In 2018, the modified KNN algorithm was validated in the task by Jo [7]. The journal article which describes in detail the modified KNN algorithm and its application to the text segmentation is finally prepared for its publication by Jo [9]. It was pointed out that the text segmentation is a domain dependent classification, and it is necessary to tag a domain to an input text automatically.

Let us explore the previous works on the table matching algorithm as the early case of encoding a text into a table. In 2008, the table based matching was proposed as an approach to the text classification as the initial case of encoding a text into a table, instead of a numerical vector [1]. In 2011, the table based matching algorithm was registered as a patent by Jo [2]. In 2015, it was upgraded into its more reliable and stable version by Jo [3]. The similarity metric between two tables is provided by the above previous works for modifying the standard KNN algorithm and its variants.

Let us mention the differences of this research from the previous works which are explored above. The text classification which exists as a single system in [9] is applied to the domain selection in the text segmentation. By introducing the text classification to the text segmentation, we automate tagging a domain to a text. The modified versions of KNN variants become more advanced ones, compared with the table based matching algorithm. In the next research, we will consider introducing text segmentation to text classification as the reverse case of this research.

III. PROPOSED WORKS

This section is concerned with what is proposed in this research. In Section III-A, we describe the process of encoding a text into a table and the proposed similarity metric. In Section III-B, we describe the table based version of KNN algorithm. In Section III-C, we present the system architecture of the text segmentation system. In Section III-D, we present the system architecture of its extended version by connecting it with the text categorization system.

A. Encoding Process

This section is concerned with the table as the text representation. It is defined as a set of entries, each of which consists of an element and its weight. The table which represents a text consists of entries, each of which is a pair of a word and its weight in the text. The similarity between tables is computed based on shared words as the semantic similarity between texts. This section is intended to describe the process of encoding a text into a table and the process of computing the similarity between tables.

The process of encoding a text into a table is illustrated in Figure 1. It is indexed into a list of words. The weights of words in the text are computed in the text-word weighting; the word weight in the text is a relationship between a word and a text. The entries with their lower weights are removed for downsizing the table. Refer to [1] for studying the entre process in more detail.

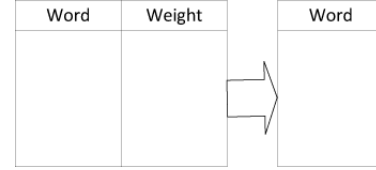


Figure 1. Process of Encoding Texts into Tables

Let us mention the function of a table which maps into a set of words as shown in Figure 2. The table which represents a text is expressed as entry set as shown in equation (1),

$$T = \{(t_1, w_1), (t_2, w_2), \dots, (t_{|T|}, w_{|T|})\} \quad (1)$$

where t_i is a word, and w_i is the weight of the word, t_i , in the text. The function for generating a list of words is expressed as equation (2),

$$F(T) = \{t_1, t_2, \dots, t_{|T|}\} \quad (2)$$

The function is the operation which takes only words without their weights as a set. The operation is applied to computing the similarity between tables which represent texts.

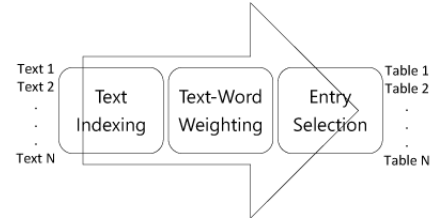


Figure 2. Conversion of Table into Word Set

Let us mention the process of computing the similarity between two tables as the similarity between two texts. The tables which represent texts are notated by the two

sets: $D_1 = \{(t_{11}, w_{11}), (t_{12}, w_{12}), \dots, (t_{1|D_1|}, w_{1|D_1|})\}$ and $D_2 = \{(t_{21}, w_{21}), (t_{22}, w_{22}), \dots, (t_{2|D_2|}, w_{2|D_2|})\}$. The intersection between the two sets which are generated by applying the function, $F(\cdot)$, by equation (3),

$$F(D_1) \cap F(D_2) = \{t_{s1}, t_{s2}, \dots, t_{s|F(D_1) \cap F(D_2)|}\} \quad (3)$$

and the table with entries each of which consists of a shared word, its weight from the table, D_1 , its weight from the table, D_2 , is constructed based on equation (3), as expressed in equation (4),

$$SD_{12} = \{(t_{s1}, w_{s1}^1, w_{s1}^2), (t_{s2}, w_{s2}^1, w_{s2}^2), \dots, (t_{s|F(D_1) \cap F(D_2)|}, w_{s|F(D_1) \cap F(D_2)|}^1, w_{s|F(D_1) \cap F(D_2)|}^2)\} \quad (4)$$

The similarity between the two tables, D_1 and D_2 , is computed by equation (5),

$$sim(D_1, D_2) = \frac{2(\sum_{i=1}^{|F(D_1) \cap F(D_2)|} w_{si}^1 + \sum_{i=1}^{|F(D_1) \cap F(D_2)|} w_{si}^2)}{\sum_{i=1}^{|D_1|} w_{1i} + \sum_{i=1}^{|D_2|} w_{2i}}. \quad (5)$$

The value which is computed by equation (5), is always given as a normalized value between zero and one, as shown in equation (6),

$$0 \leq sim(D_1, D_2) \leq 1. \quad (6)$$

Let us make some remarks on the encoding process and the computation of the similarity between texts. A text is encoded into a table by the text indexing, the word weighting, and the entry selection. A table is expressed as a set of entries, and a table is filtered into a set of words by the defined function. The similarity between tables is computed based on shared words by equation (5). In the future research, we consider representing a text into multiple tables.

B. Table based KNN Algorithm

This section is concerned with the table based KNN algorithm. It was initially proposed as the approach to the text classification by Jo in 2015 [3]. The similarity metric between tables which was described in the previous section is used for computing the similarity between a novice table and a training example. The labels of its nearest neighbors are voted with their identical weights for deciding the label of a novice input. This section is intended to describe the initial version of KNN algorithm from which its variants are derived.

The process of computing the similarity between a novice item and a training example is illustrated in Figure 3. The labeled words which are gathered as samples are encoded into tables, and they are notated by a set $Tr = \{T_1, T_2, \dots, T_N\}$. A novice word is encoded into a table notated by T . Its similarities with the tables which represent the sample words are computed by equation (5),

as $sim(T, T_1), sim(T, T_2), \dots, sim(T, T_N)$. Some training examples which are most similar as the novice input are selected as its nearest neighbors.

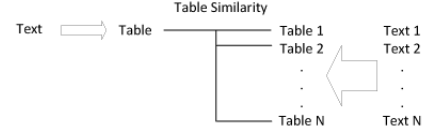


Figure 3. Similarities of Novice Input with Training Examples

In Figure 4, the process of selecting nearest neighbors by the ranking selection is illustrated. The training examples are sorted by the descending order of their similarities, after computing their similarities with a novice example. The training examples with their higher similarities with a novice example are selected as its nearest neighbors, as expressed in equation (7),

$$Nr = Select_k(Tr, T), Nr \subseteq Tr \quad (7)$$

The set of nearest neighbors, Nr , is composed with elements as expressed in equation (8),

$$Nr = \{T_1^{near}, T_2^{near}, \dots, T_k^{near}\} \quad (8)$$

The elements in the set, Nr , are used for voting their labels in the next step.

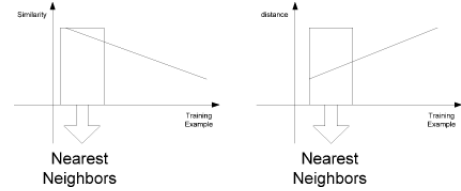


Figure 4. Selection of Most Similar or Least Distant Training Examples as Nearest Neighbors

The process of voting the labels of nearest neighbors for decoding the label of a novice word is illustrated in Figure 5. The predefined categories are notated by $c_1, c_2, \dots, c_m$, and the label of a nearest neighbor is notated by $c_j = label(T_i^{near})$. The number of nearest neighbors which belong to the category, c_j , is notated by $Count(Nr, c_j)$. The label of the novice item, T , is decided by equation (9),

$$label(T) = \arg\max_{j=1}^m Count(Nr, c_j) \quad (9)$$

The process of deciding the label of a novice item by equation (9), is called voting.

Let us make some remarks on the table based KNN algorithm. It computes the similarities of a novice item with the training examples. The training examples with their highest similarities with the novice item are selected as its nearest neighbors. The label of the novice item is decided by voting the labels of its nearest neighbors. The ranking

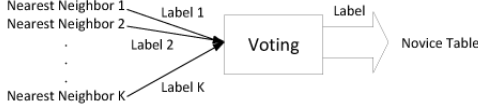


Figure 5. Voting of Labels of Nearest Neighbors for Deciding Label of Novice Input

selection is adopted for selecting the nearest neighbors from training examples, in this version.

C. System Architecture

This section is concerned with the text segmentation system which adopts the table based KNN algorithm. In Section III-B, we described the proposed version of KNN algorithm as the approach to the text segmentation. It is viewed as a binary classification which classifies a paragraph pair into boundary or continuance. This section is intended to describe the text segmentation system with respect to its function and its architecture.

The process of gathering sample paragraph pairs and classifying a novice one, is illustrated in Figure 6. Because even a same paragraph pair may be classified differently depending on the domain, the task is called domain dependent classification. Sample paragraph pairs are gathered domain by domain, and each pair is labeled with boundary or continuance. The paragraph pairs are taken from texts which are tagged with their same domain, each paragraph pair is classified into boundary or continuance. Sample paragraph pairs which are labeled with one of the two categories are encoded into tables in each domain.

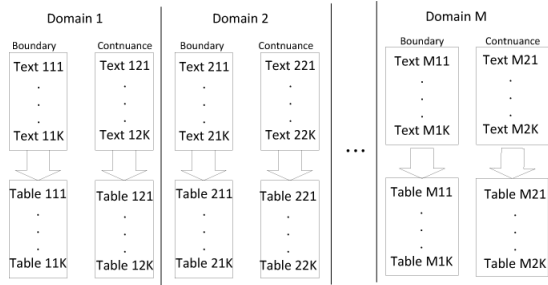


Figure 6. Preparation of Training Examples

The entire architecture of the proposed text segmentation system is illustrated in Figure 7. A text is given as the input, and adjacent paragraph pairs are extracted by partitioning the text into paragraphs and slide the two sized window on them. The sample paragraph pairs in the boundary group and the continuance group and ones which extracted from the text are mapped into tables in the encoding module. The paragraph pairs from the text are classified into one of the two categories in the similarity computation module and the voting module. A boundary is put between paragraphs in

each pair which is classified into boundary in the text, and it is partitioned into subtexts by the boundary.

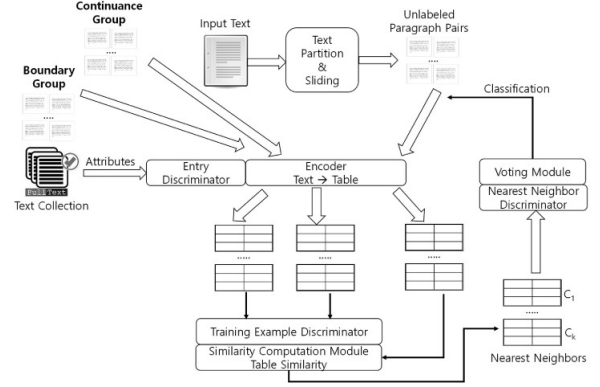


Figure 7. System Architecture

The execution flow of the text segmentation system is illustrated in Figure 8. Texts with same domain are initially collected, adjacent paragraph pairs are extracted from them, and the paragraph pairs are labeled with boundary or continuance. From an input text, adjacent paragraph pairs are extracted, and are encoded into tables, together with the sample paragraph pairs. The adjacent paragraph pairs are classified into boundary or continuance, and there are two groups of paragraph pairs: boundary group and continuance group. The boundary is marked between paragraphs in each pair in the boundary group, and text is partitioned by the marked boundaries into subtexts as the final output of this system.

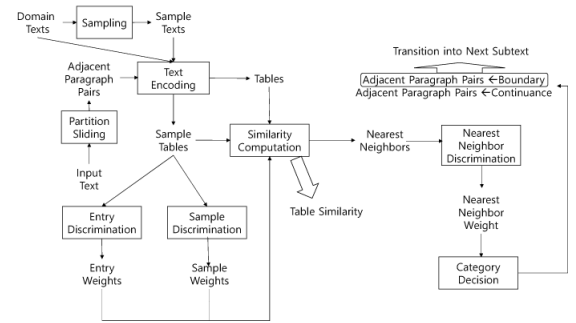


Figure 8. System Flow

Let us make some remarks on the proposed system which is illustrated in Figure 7 as its architecture. The text segmentation is defined as the binary classification where each adjacent paragraph pair is classified into boundary or continuance. The paragraph pairs are encoded into tables instead of numerical vectors, and they are classified into directly. The input text is partitioned into subtexts by the boundary which is put between paragraphs which are classified into boundary. The subtexts as the semantic division of the input text are treated as independent texts.

D. Connection with Text Classification

This section is concerned with the connection of the text segmentation with the text classification. In the previous section, we mentioned the text segmentation system as an instance of domain dependent classification. The domain is automatically decided for nominating a classifier by connecting the keyword with the text classification. The KNN algorithm which was described in Section III-B is used for implementing the text classification. This section is intended to describe the connection of the text segmentation system with the text classification for automating the decision of the domain.

The system architecture of the text categorization system which consists of the encoding module, the similarity computation module, and the voting module is illustrated in Figure 9. The encoding module is for encoding texts into tables by the process which is described in Section III-A. The similarity computation module is for computing the similarities between tables which represent a novice text and a sample text and selecting ones with their highest similarities as the nearest neighbors. The voting module is for deciding the label of the novice item by voting the labels of the nearest neighbors. This system used for automating deciding the domain of the input text.

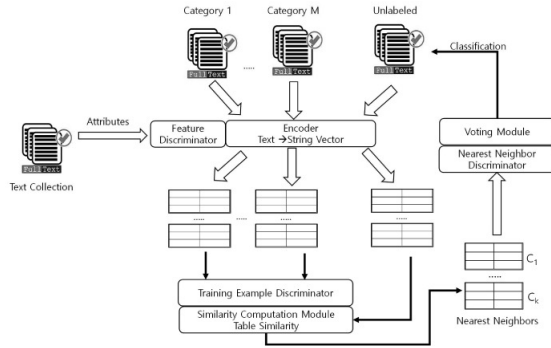


Figure 9. Text Categorization System

The connection of the text segmentation with the text classification is illustrated in Figure 10. The texts each of which is labeled with its own domain are prepared as the sample texts, and the approach to the text classification is trained with them. A domain is assigned to a novice text by classifying it into a domain automatically. The novice text is provided with its domain to the text segmentation system which is described in Section III-C. The role of the text classification system is to nominate the classifier which corresponds to the domain for the text segmentation, automatically.

The process of executing the text segmentation, connecting it with the text classification is illustrated in Figure 11. The sample texts each of which is labeled with its own domain, and sample paragraph pairs each of which

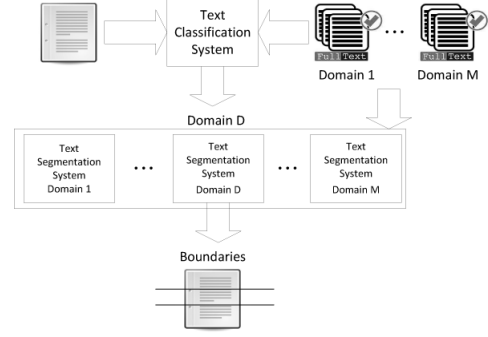


Figure 10. Text Segmentation System connected with Text categorization

is labeled with boundary and continuance are prepared in each domain. The novice text is classified into one among the predefined domains, and the classifier which correspond to the domain is nominated. Adjacent paragraph pairs are generated by sliding the two sized window on the paragraph list, and each paragraph pair is classified into boundary or continuance. Subtexts which are generated by segmenting the text following the boundaries are generated as the final output; the domain which is assigned to the input text is the guide for selecting a classifier.

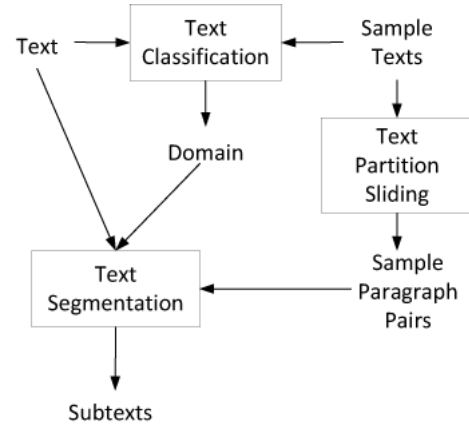


Figure 11. System Flow of Text Segmentation connected with Text Categorization

Let us make some remarks on the connection of the text segmentation with the text classification. The text classification is applied to the automatic decision of the input text domain. The role of text classification is to decide the domain for nominating a classifier, and the role of the text segmentation is to partition the entire text into subtexts based on its content. In the execution flow, the input text is classified into a domain, paragraph pairs are generated by sliding the two sized window, and each paragraph pair is classified into boundary or continuance. We consider connecting the text classification as the main task the text segmentation as the opposite to what is proposed in this

research.

IV. CONCLUSION

Let us mention the significances of this research as the conclusion. We modify the KNN algorithm into the table based version. We design the text segmentation system by adopting the proposed KNN algorithm. We connect the system with the text categorization system for automating the domain dependent classification. In the next research, we will implement the text segmentation system as a real system.

Let us mention remaining tasks for improving this research. We may define and characterize mathematically more operations on tables, in addition to the similarity metric between two tables. Paragraph pairs and texts may be encoded into compound representations, each of which consists of more than one structured form. We may modify the KNN variants, other machine learning algorithms, and deep learning algorithms into table-based versions as the approaches to the text segmentation and the text classification. The text segmentation system which includes the text classification module may be implemented as a real program or a library by adopting the table based KNN algorithm.

REFERENCES

- [1] T. Jo, "Table based Matching Algorithm for Soft Categorization of News Articles in Reuter 21578", 875-882, Journal of Korea Multimedia Society, 11, 2008.
- [2] T. Jo, "Device and Method for Categorizing Electronic Document Automatically", 10-2009-0041272, 10-1071495, 2011.
- [3] T. Jo, "Normalized Table Matching Algorithm as Approach to Text Categorization", 839-849, Soft Computing, 19, 2015.
- [4] T. Jo, "Table based KNN for Article Classification", 271-276, The Proceedings of 19th International Conference on Artificial Intelligence, 2017.
- [5] T. Jo, "Content based Segmentation of Texts using Table based KNN", 27-32, The Proceedings of 16th International Conference on Advances in Information and Knowledge Engineering, 2017.
- [6] T. Jo, "Modification into Table based K Nearest Neighbor for News Article Classification", 49-50, The Proceedings of 1st International Conference on Advanced Engineering and ICT-Convergence, 2018.
- [7] T. Jo, "Using Table based Version of K Nearest Neighbor for Segmenting News Articles by their Contents", 62-64, The Proceedings of 1st International Conference on Advanced Engineering and ICT-Convergence, 2018.
- [8] T. Jo, "Applying Table based K Nearest Neighbor to Text Classification", in Preparation, 2023.
- [9] T. Jo, "Segmenting Long Texts Automatically by Table based K Nearest Neighbor", in Preparation, 2023.